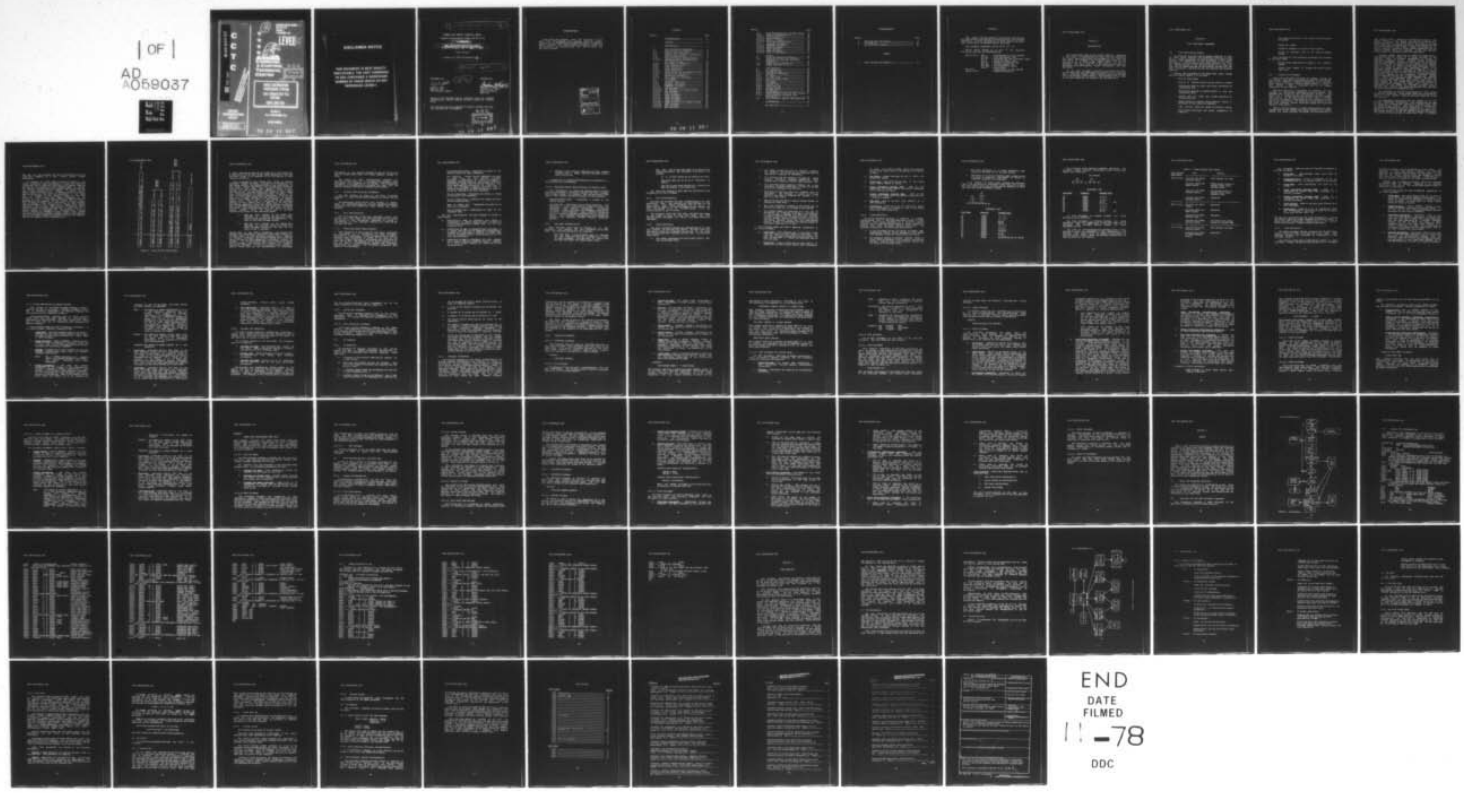
AD-A059 037 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON D C F/G 5/2
NMCS INFORMATION PROCESSING SYSTEM 360 FORMATTED FILE SYSTEM (N--ETC(U)
SEP 78
UNCLASSIFIED CCTC-CSM-UM-15-78-VOL-2 NL

| OF |
AD
A059037



END
DATE
FILMED
11-78
DDC

CCTC

COMPUTER SYSTEM MANUAL
CSM UM 15-78
VOLUME II
1 SEPTEMBER 1978

LEVEL III

COMMAND & CONTROL TECHNICAL CENTER

NMCS INFORMATION PROCESSING SYSTEM

360 FORMATTED FILE SYSTEM

(NIPS 360 FFS)

VOLUME II

FILE STRUCTURING (FS)

USERS MANUAL

DEFENSE COMMUNICATIONS AGENCY

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

(14) CCTC-CSM-UM-15-78-VOL-2

COMMAND AND CONTROL TECHNICAL CENTER

Computer System Manual Number CSM UM 15-78

(11) 1 Sep[redacted]78

(6) NMCS INFORMATION PROCESSING SYSTEM
360 FORMATTED FILE SYSTEM (NIPS 360 FFS).

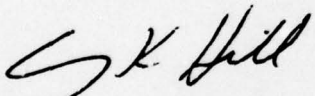
Users Manual,

Volume II.- File Structuring (FS).

(9) Computer systems manual.

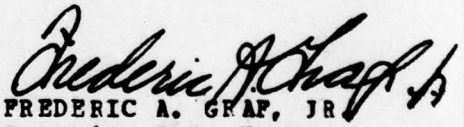
SUBMITTED BY:

(12) 78p.



CRAIG K. HILL
Captain, USA
CCTC Project Officer

APPROVED BY:



FREDERIC A. GRAF, JR
Captain, U.S. Navy
Deputy Director,
NMCS ADP

Copies of this document may be obtained from the Defense Documentation Center, Cameron Station, Alexandria, Virginia 22314

This document has been approved for public release and sale; its distribution is unlimited.

409 658

78 08 11 007

ACKNOWLEDGMENT

This manual was prepared under the direction of the Chief for Programming with general technical support provided by the International Business Machines Corporation under contracts DCA 100-67-C-0062, DCA 100-69-C-0029, DCA 100-70-C-0031, DCA 100-70-C-0080, DCA 100-71-C-0047, and DCA 100-77-C-0065.

CONTENTS

ILLUSTRATIONS

TABLES

ABSTRACT

This volume focuses primarily on detailed instructions for effectively using the File Structuring and File Revision Components. A brief introduction to these system components and a description of expected output are given.

This document supersedes CSM UM 15-74, Vol. II.

CSM UM 15-78, Volume II is part of the following additional NIPS 360 FFS documentation:

CSM UM 15-78	Vol I	- Introduction to File Concepts
	Vol III	- File Maintenance (FM)
	Vol IV	- Retrieval and Sort Processor (RASP)
	Vol V	- Output Processor (OP)
	Vol VI	- Terminal Processing (TP)
	Vol VII	- Utility Support (UT)
	Vol VIII	- Job Preparation Manual
	Vol IX	- Error Codes
TR 54-78		- Installation of NIPS 360 FFS
CSM GD 15-78		- General Description

FILE STRUCTURING (FS)

Section 1

INTRODUCTION

File Structuring (FS) is the initial phase in designing a new file. In structuring the file, the user determines the type of data which should be included in the file and the order of grouping. The structured data table is then referred to as a File Format Table (FFT). An FFT is generated for each file and is the basic outline used for performing file maintenance, output processing, and data retrieval.

FS can also be used to describe a non-NIPS data file. In this mode the user describes the contents of an existing file, including record control fields, record type fields and nonaccessible FILLER fields. The resulting FFT is used by the QUIP component for querying the non-NIPS file.

Section 2

FILE STRUCTURING COMPONENT

2.1 File Structuring Program

For input the File Structuring program requires a deck of cards which contains the parameters defining the file format. These parameters are used to develop a table which provides the definition of the file format to the rest of the system. The File Format Table (FFT) contains not only the specifications of the internal format of the file, but also partial specifications of the external formats and the data conversion required between the external and internal formats.

Briefly, the parameters in the input deck which define the file record format to FS include:

File ID (File Name)

Record ID (Record Control Group) fields or groups

Subroutines used for input and output conversion of field/group data

Subroutines applied to keyword fields to scan the transaction data.

Tables used for input and output conversion of field/group data

Tables applied to values within keyword fields to determine if the value is a keyword.

Edit control words for output of fields or groups

Field names with sizes and modes (alphabetic or numeric)

Group names with list of all fields included within the group.

Labels for output.

Fields or groups to be used as file indexes.

Fields or variable sets to be used as keyword fields.

For a non-NIPS FS, the following parameters are included in the input deck:

Record format definition to assign a set sequence number.

Record type fields to contain the record format type code.

2.2 General File Concepts

Files have a definite structure or pattern called the file format. When data is grouped according to file format, each group contains a complete description of an activity, event, person, or thing. Each of these groups is called a file record. Thus, a Formatted File is an ordered collection of file records.

The file record is a collection of elements of data arranged in the pattern specified by the file format. The smallest unit of data is the field/record element. Each field has a defined length and contains only one specific type of data. If the data content of the field is fixed, the field is labeled a fixed field and will appear only once within the file record. In addition to fixed fields, a file may have periodic fields which may appear more than once within a file record.

Within the file record, a closer relationship can exist between some of the fields than exists between other fields. Fields are often reported, retrieved, and manipulated as a

unit called a group. Groups are collections of two or more fields within a file record and may be considered as a single data entity. The fields within the group do not lose their individual identities because of this grouping and may be treated like any other field. Groups are categorized as either periodic groups or fixed groups, depending upon the types of fields which are grouped. Fixed and periodic fields cannot be included within the same group.

Periodic fields may occur any number of times within a file record and the change of a value recorded in one periodic field is usually accompanied by changes in associated periodic fields. Periodic fields containing related information are grouped into units called subsets. When periodic data is added to a file record, it is usually added in units of subsets rather than individual fields. This is not meant to imply that individual fields of an existing subset cannot be corrected or updated. It is the subset structure which is specified by the file format, while provision is made for the entry of as many identically structured subsets as are necessary. Periodic subsets having identical formats are grouped together into periodic sets. Thus, each periodic subset within a periodic set contains the same field in the same order, with only the data content of the fields varying. Periodic subsets of other formats are grouped into other sets. There is no limitation on the number of periodic subsets within a periodic set. However, the number of periodic sets per file record is limited to a maximum of 255.

The collection of all the fixed fields of a file record is called a fixed set. Since fixed fields are nonrepetitive within a file record, there is only one fixed set in any file record.

One additional type of data may be placed in a file record: commentary information. This unformatted data may be placed in a variable set, defined as a single field with an unrestricted length. This data may be accessed for retrieval by employing one of two capabilities. The first method is to use the CONTAINS operator at retrieval time. The second method is to use the Keyword Indexing capability. If neither of these methods is used, data from a variable

set may not be retrieved, but may be displayed along with other data elements of a file record during output processing.

An example of a file record is shown in figure 1. The format illustrated would correspond to a data file which was structured with a fixed set containing data element names FA through FN, one periodic set with element names 1A through 1F, another periodic set with element names 2A through 2H, and a single variable set for remarks. A record control group, which contains data to identify each file record, must be defined as the first data element(s) of the fixed set. It appears in the graphic layout as the term RECORD ID. A data record will appear internally to the machine as one or more logical records, each of which is prefixed with the contents of the record control group. This technique causes the association of many internal logical records to be classed as a single file record externally for the user. Specifically, an internal logical record is used to contain the data for a fixed set or a single subset of a periodic set. The portions of the file record labeled as RECORD ID also contain some system generated and maintained data elements used to differentiate the logical records among the periodic sets and the sequence numbers of subsets within a periodic set.

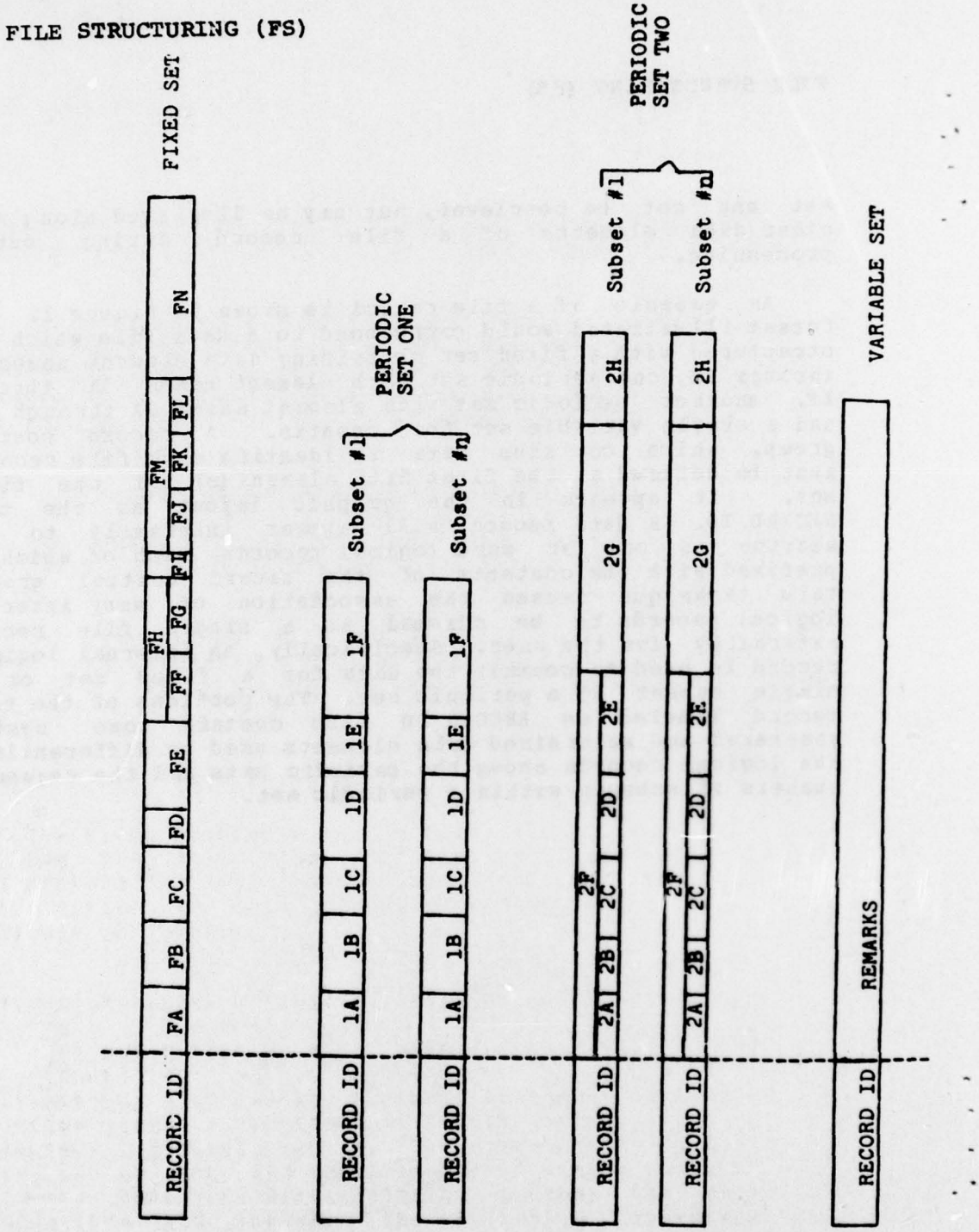


Figure 1. NIPS 360 FFS FILE RECORD

A more detailed account of the format of a file record and a description of the file format table generated by the file structure appear in the appendix of the Introduction to File Concepts (Volume I).

Data is stored in the system data files in several different formats or modes. Although it is possible to use the system without full understanding of the intricacies of data mode, system efficiency can be improved by correct specification and usage. Four specific modes of data storage are used by the system: Alphameric, Decimal, Binary and Coordinate. The user explicitly controls these modes when using alphameric or coordinate forms. However, in the case of numeric data, the system effectively controls the mode and the user assigns the mode implicitly. An overall generic term, NUMER, is used in the system and implies numeric data. This term is described in the following sections of this volume and is used only by the File Structuring component. Note that neither binary nor decimal may be specified directly by the user except for subroutine attributes. The system assigns one of two modes to every numeric (NUMER) field; the binary mode is provided automatically except in the following cases:

a. When the field defined by the NUMER mode statement is a part of the record or subset control group (record key or identifier), the mode of the field becomes decimal.

b. When the field defined by the NUMER mode statement is included within the range of a GROUP statement definition, the mode of the field is set to decimal.

During most of the system processing, the analyst need consider only alphameric, coordinate, and numeric (binary or decimal) modes. In some instances, however, he is required to directly reference the internal system mode in precise terms, and must specify alphameric, coordinate, binary, or decimal. In all cases where mode is not explicitly stated by the analyst, the system will provide the automatic conversion necessary to "match" or adjust modes correctly. Only in cases such as the specification of subroutine

attributes is the analyst required to use one of the four specific modes rather than the three generic classes of data.

The types of mode specification required are specifically discussed on a statement-by-statement basis throughout the remainder of this volume. This preliminary discussion is intended only to acquaint the user with the general concepts involved in internal data formats.

2.3 General File Structure Statements

The user submits as input to the file structure component a deck of statements defining the format for his file.

The subsections which follow will discuss in general terms the statements used by the file structure language. Section 2.4 will then define the exact format and syntax required.

2.3.1 File Identification

The first statement in the file structure source deck defines the file name. The data file name must be unique and conform to name conventions described in Volume I, Introduction to File Concepts section 2.6.3 NIPS 360 FFS Language contents.

2.3.2 Subroutine/Table Identification

Each subroutine or table used for data value conversion is identified by a subroutine or table statement. Furthermore, a subroutine or table statement must be used to describe each subroutine or table to be used by any Secondary Indexing function. The operator may be either the term SUB or TAB. The optional use of either operator is to aid the user in distinguishing between his conversion subroutines and tables. The following parameters must appear in the statement:

Subroutine/Table Name - Identifies a module on the system library for data conversion.

Use Function - Identifies the conversion of input or output data. If this SUB/TAB statement describes a subroutine or table to be used by a Secondary Indexing function, then either CONVERT or ANALYZE must appear here as the Use Function. This entry describes the function of the subroutine or table for Secondary Indexing purposes only. A detailed description of these functions is found in Volume IV, Retrieval and Sort Processor.

Size of Input Data - Indicates the length in bytes of the data to be converted.

Size of Output Data - Indicates the length in bytes of the converted data.

Mode of Input Data - Identifies the mode of the data to be converted.

Mode of Output Data - Identifies the mode of the converted data.

The mode specification for data elements is stated in exact terms:

a. ALPHA mode is used to reference data which is represented in EBCDIC notation form. Elements in a file record defined as ALPHA mode are of this characteristic.

b. BINARY mode is used to reference data contained in a binary word (32 bits of numerical significance). Elements in a file record defined as NUMBR mode (and not included within a group definition) are of this characteristic.

c. COORD mode is used to reference any data element with a system internal coordinate format. Elements in a file record defined as COORD are of this characteristic.

3. DECIMAL mode is used to reference any data element defined with a NUMER mode and included within a group definition.

A maximum of 50 subroutines/tables may be defined for a file during file structuring.

2.3.2.1 Subroutine/Table Identification for Keyword Usage

When subroutines or tables are applied against keyword fields they perform a different function (see SUB Statement for Keyword Usage for details) but the same basic rules apply. The following parameters appear on the statement:

Subroutine/Table Name - Identifies a module on the system library.

Use Functions - Identifies one of three functions. STOP indicates a table containing all words that are to be eliminated from keyword processing. DICTIONARY indicates a table containing all allowable keywords for the field. SCAN indicates a user provided subroutine that is to extract keyword candidates from a transaction record. For a detailed description of these tables see Volume I, File Concepts, Keyword Indexing Capability section.

2.3.3 Edit Mask Identification

Each user-edit mask must be defined by an edit statement. The operator for this statement is the term EDIT. The following parameters are required:

Edit Mask Name - A user-supplied name to identify the edit mask when used as a parameter in a field or group statement. The name must be unique for the file and follow the system names rules.

Edit Mask - The actual edit mask to be used during output conversion. The following limitations are imposed:

Up to 50 edit masks may be defined per file.

Each edit mask may be up to 69 characters in length.

The sum of edit mask lengths for a single file must not exceed 1,000 characters.

The rules for writing an edit mask are specified in the Introduction to File Concepts.

2.3.4 File Classification

The user may identify the file classification at file structuring time. The operator for this statement is the term CLASS. This single parameter may be up to 32 characters in length. If this statement is not present, the file structured with no classification, i.e., the classification is set to blanks.

The statement types SUB, TAB, EDIT, and CLASS may appear in any order as long as they come after the file identification statement and before any FIELD definition statement.

2.3.5 Field Definition

The field statement defines the characteristics of each data element in the data file record. The user writes the statements, defining in sequence the entries of the fixed set, Periodic Set One, Periodic Set Two, etc. The following rules apply:

a. The fields containing the file record control data must be defined first.

b. All fields in each set must be defined together; for example, all fields for the fixed set must be defined in a sequence of statements.

c. If a periodic set has fields to be used for subset control (standard language capability), these fields must be defined first in that periodic set.

d. If a variable length field is defined for a set (standard language capability), it must be defined by the last statement for that set.

e. When defining periodic sets, set numbers must be specified. For example, a record with three periodic sets must be assigned the numbers 1, 2, and 3, in this order.

f. Record control and subset control fields should be defined as ALPHA mode.

g. Up to and including a maximum of 100 fields may be defined for each set of the user's file record.

h. The maximum number of characters in a set may be determined by referring to Volume I, Introduction to File Concepts, Appendix A.2, Data File Records. By knowing the size of the record control field and largest subset control field, the user can add up the system overhead bytes and the user-defined control fields to determine the maximum number of noncontrol characters in a set.

The following terms are used to describe parameters in the FIELD statement:

a. Field Name - User-assigned name to the field which will be used to access specific record data. The name must be unique within the file and follow the same conventions of length and character mode as the file name.

b. Field Size - Size in bytes for the data item as it appears in the data record (after input conversion,

if used). For numeric fields, which are stored as binary words (four bytes), the external form length is used.

c. Set Number - Number defining the set to which the field will belong.

d. Field Mode - Term defining the mode of the field (e.g., ALPHA, NUMER, or COORD).

e. Input Conversion Routine Name - Name of the subroutine or table which was earlier described with a Sub/Tab statement.

f. Output Conversion Routine Name - Name of the subroutine/ table which was earlier described with a Sub/Tab statement.

g. Edit Name - Name of the edit mask defined by an edit statement.

h. Field Label - Label of up to 69 characters which may be assigned to the field in lieu of the actual field name when using QUIP.

2.3.6 Group Definition

The group statement redefines a sequence of adjacent fields/groups with a new name which may be used as a data reference. This group statement is a powerful logical operator which allows the manipulation of larger segments of related data, thus increasing machine efficiency. The following conventions and rules must be followed:

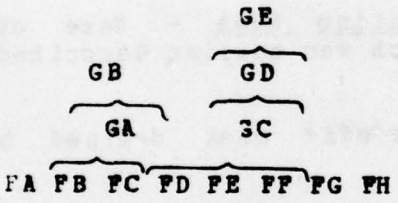
a. In the statement string, the group statement must immediately follow those statements which define data elements including those within the group.

b. All elements (fields or groups) defined within a group must be adjacent to one another; none may be omitted. Control fields and noncontrol fields may not be part of the same group.

c. All items appearing in a group definition must belong to the same set of the data record.

d. Definition of a group including other groups within its range is permitted. The number of levels or "nesting" permitted is unlimited.

As an example of these rules, consider the following format and the pseudo FS statements which define the format. Note the nesting which may be done and the sequence in which the field and group statements appear.



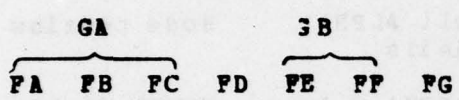
STATEMENT LIST

Item Name	Operator	Operand List
FA	FIELD	--- ----
FB	FIELD	--- ----
FC	FIELD	--- ----
GA	GROUP	FB,FC
GB	GROUP	FA,GA
FD	FIELD	--- ----
FE	FIELD	-------
FF	FIELD	-------
GC	GROUP	FD,FE,FF
FG	FIELD	-------
GD	GROUP	GC,FG
FH	FIELD	-------
GE	GROUP	FD,FE,FF,FG,FH (or GD,FH)

FILE STRUCTURING (FS)

The following format depicts a desired structure. The statement source includes intentional errors illustrating common rule violations.

Set Format



STATEMENT LIST

Item Name	Operator	Operand List	
FA	FIELD	-------	
FB	FIELD	-------	
FC	FIELD	-------	
GA	GROUP	FA,FC	(1)
FD	FIELD	-------	
FE	FIELD	-------	
FF	FIELD	-------	
FG	FIELD	-------	
GB	GROUP	FE,FF	(2)

(1) This statement is invalid because the group definition omits field FB.

(2) This statement is invalid because the group definition does not immediately follow the items that it redefines. This statement should precede the field statement for FG.

The mode of the fields/groups included within a group definition must be considered. Some combinations are invalid, while others cause the system to redesignate an element's mode. Table 1 shows the possible combinations of mode grouping and the results.

Table 1. Mode Grouping and Results

User-Defined Group Mode	Case	Results
ALPHA Group	Contains all ALPHA defined fields	Mode remains the same
	Contains ALPHA and NUMER fields	NUMER fields change to decimal mode; group remains ALPHA
	Contains all NUMER defined fields	All fields change to decimal mode, group remains ALPHA
	Contains any COORD defined fields	Rejected!
NUMER Group	Contains all ALPHA fields	Rejected!
	Contains ALPHA and NUMER fields	Rejected!
	Contains any COORD defined field	Rejected!
	Contains all NUMER defined fields	All fields and groups change to decimal mode
COORD Group	Contains all COORD defined fields	Mode remains the same
	Contains any ALPHA or NUMER defined fields	Rejected!

The following terms are used to describe parameters in a GROUP statement:

a. Group Name - User-assigned name which must be unique to the file.

b. Field/Group List - A list, in sequence, of all the fields/groups to be defined within the new group.

c. Group Mode - Term designating the mode of the group.

d. Input Conversion Routine Name - Name of a subroutine or table which was described in a Sub/Tab statement.

e. Output Conversion Routine Name - Name of a subroutine or table which was described in a Sub/Tab statement.

f. Group Editing - Name of the edit mask defined by an edit statement.

g. Group Label - Label of up to 69 characters which may be assigned to the group in lieu of the actual group name during a QUIP run.

The user may optionally use the group statements to provide additional names for single elements in the data record. A group statement referencing a single field in its operand string will provide a limited synonym capability.

2.3.7 Index Definition

The INDEX statement defines an element of the data file, either a field or group, to be used as a file index. File indexing concepts are discussed in Introduction to File Concepts, Volume I.

Any field or group that is defined by a field or group statement, except variable-length fields and not exceeding

30 bytes in length, may be designated as a file index. One exception to this rule exists, however. If the field specified is a record control field, and the length is the same as the record control field, the attempt to specify that field as an index will be rejected. Cross-indexing a file by Record ID is redundant and would be inefficient.

Indexes may be fixed or periodic. There is no maximum or minimum number of indexes. Indexing is an optional capability, and any or all non-variable-length fields or groups may be specified.

The following terms are used to describe parameters in the INDEX statement:

a. Field Name - The user-assigned name of the field or group which will access specific record data. The field or group must be defined by a valid FIELD or GROUP statement. An index field or group cannot exceed 30 bytes in length.

b. Index Operation - (ADD or DELETE) - Default is ADD. DELETE is used to eliminate an existing index. It is for use by the Index Specification function performed by the Utility, UTMDXSPC.

c. Conversion Subroutine - (Optional) This is the name of a user-provided subroutine to convert data from data file form to index form. This subroutine must have been identified by a SUB/TAB statement during the run in which the index is being specified, and it must have been defined as a CONVERT subroutine or table. If no subroutine is specified, the values for the index field will be in their internal data file format.

d. Analyzer Subroutine - (Optional) This is the name of a user-provided subroutine used to analyze and determine the qualification value(s) for any FUNCTION operator in which the indexed field/group is a parameter. It must have been defined as an ANALYZE subroutine or table on a SUB/TAB statement.

2.3.7.1 Index Definition for Keyword Fields

This format of the INDEX statement defines a field or variable set that is to become a Keyword Indexing Field. For a detailed description of Keyword Indexing see Introduction to File Concepts, Volume I.

Any variable field, variable set, or fixed periodic alpha field that has been defined for the file may be specified as a keyword field, the only restriction being that a field may not be defined for both secondary and keyword indexing.

The following terms are used to describe parameters in the INDEX statement defining a keyword field:

a. Field Name - The user-assigned name of the field or variable set that will access specific record data. The field or set must be defined by a valie FIELD or VSET statement.

b. Index Operation - (ADD or DELETE) - Default is ADD. DELETE is used to eliminate an existing index and is used only by the Index Specification function performed by the Utility, UTNDXSPC.

c. KEYWORD - Designation of this statement as one that defines a keyword field rather than the standard Secondary Index Field.

NOTE: For a DELETE operation on a keyword field, the KEYWORD parameter is optional. The delete operation will occur regardless.

d. Hyphen Designation - The user can control processing of the hyphen by the system's scan routine. A hyphen is always treated as a word separator if it follows a word separator character. If it is followed by a text character, one of four options can be applied. TEXT is the default option which means the hyphen is to be treated and

retained as part of the value. The other options are DROP, RETAIN, and SEPARATE.

DROP - The hyphen is either superfluous or it is a word separator, depending upon its context with respect to adjacent characters. It is superfluous if: (1) text characters immediately precede and follow it, or (2) it is immediately preceded by a text character and followed by one or more blank characters and a text character. In this case, blanks are considered to be superfluous. If neither of these cases apply, the hyphen is treated as a word separator. In any case, the hyphen is dropped.

RETAIN - The hyphen is subject to the same rules as the DROP option, except that in the two cases where it is considered superfluous it is retained in the value as a text character.

SEPARATE- The hyphen is always treated as a word separator.

e. Stop Table - (Optional) This is the name of the user provided table that is to be applied against the values in this field. If a value is found in the table, it is to be eliminated from any further consideration as a keyword. If a value is not present in the table, or if no table were specified, the value becomes a keyword candidate. This table must have been identified by a SUB/TAB statement and have been defined as a STOP table.

f. Dictionary - (Optional) This is the name of the user provided table containing all nonliteral values from this field that may become keywords. This table must have been identified by a SUB/TAB statement and have been defined as a DICTIONARY. If no dictionary is specified, all values in the field, unless some were eliminated by a Stop Table,

become keywords. Literal words always become keywords.

g. Scan Subroutine - (Optional) This is the name of the user provided subroutine that is to be invoked to process the keyword field. This subroutine must have been identified by a SUB/TAB statement and have been identified as a SCAN subroutine. If no subroutine is specified, the system's scan routine will determine what qualifies as a value for keyword selection. See Volume I; File Concepts, User Scan Subroutine section, for a generalized description of the system scan routine.

2.3.8 Variable Set Definition

The VSET statement defines a variable set containing a single field which may be used to store commentary data in textual form. There is no practical limit to the amount of data which may be placed in this set.

The statement operator is the term VSET. The parameters used are as follows:

a. Variable Set Name - User-assigned name unique for the file. Name conventions defined in Volume I, Introduction to File Concepts, apply.

b. Display Size - Value expressing the size of data to be printed per line during QUIP component processing.

c. Variable Set Label - Label of up to 69 characters may be used in lieu of the name during QUIP processing.

The variable set statement(s) should appear in the source deck after all field and group statements. Since no operand is used to designate a set number, the file structure component will automatically assign the next set available. For example, if the last periodic set defined by

the user through field and group statements was 18, the first variable set would become set number 19.

2.3.9 End-of-Job Statement

The end-of-job statement appears last in the source-statement deck. It signifies the completion of file definition. It contains the operator term ENDFS and has no parameters.

2.3.10 User Commentary Statement

If the user desires commentary remarks on the source listing, this capability is provided by the NOTE statement. It may appear anywhere between the file definition and end-of-job statement. There is no limit on the number of NOTE statements which may appear in a run.

2.4 FS Language

2.4.1 Introduction

The standard FS language discussed in this section allows the user to define additional capabilities not offered with the compatible NIPS 1410 FFS language. Among these are:

a. A maximum of 255 periodic sets may be defined for a file record.

b. More than one variable set may be defined. (The sum of the number of periodic and variable sets may be up to 255.)

c. A variable length field may be defined for each set to hold commentary data.

d. A subset control group may be defined. One or more fields specified at the beginning of a periodic set

can be flagged for direct subset identification in place of sequence numbers.

e. A total of 100 fields or groups may be defined for each set.

f. A maximum of 99 fields may be defined as a group.

g. A maximum of 50 groups may be defined in a set.

h. The record control group may be as large as 244 characters.

i. The number of subsets per set is a variable and is a function of the available computer core storage and the processing function requested. No theoretical limit is established and the practical limit is large in comparison to previous systems.

j. Fields and groups may be specified as file indexes. Any field not variable in length and not system-generated may be specified as a secondary index. Any fixed-alpha field, variable field, or variable set may be specified as a keyword index. Any fixed-alpha field, variable field, or variable set may be specified as a keyword index. No fixed field may be both a secondary and keyword index. There may be as many indexes as there are user-defined fields in the data file.

2.4.2 Language Conventions

The standard language is free format as described by the general language description and detailed in Volume I, Introduction to File Concepts. Text strings are considered to be serial strings of words/characters without regard to card (record) boundaries and may be initiated in any character position. Words may not be split between cards (records); however, any number of commas/blanks may be used to separate words. The source language may be in either card or sequential data set form. Only the first 71 characters of each record are used. However, sequence

information may be provided in characters 73 through 80 and will be checked by the module. If the sequence field indicates a break in sequence, an advisory message will be issued; however, structuring will still be performed. The EBCDIC collating sequence is used for performing the sequence check; all cards of the input stream including and following the STRUCTURE card will be tested. If sequence checking is not desired, the first card of the input should contain blanks in the sequence field.

The input stream is made up of a series of statements. By definition, each statement is a series of words or terms initiated by an operator and terminated by a period. The sequence of the statements is subject to general rules which are provided in the following sections. Each statement type is discussed in the sequence of its occurrence in the input stream.

2.4.3 Language Statements

2.4.3.1 STRUCTURE Statement

This statement uses the operator STRUCTURE followed by the data file name. This statement must occur first in the input deck. It is used to establish the name of the file. File names conform to standard system naming rules as defined in Volume I, Introduction to File Concepts.

Example:

```
STRUCTURE SOFTEST.
```

2.4.3.2 SUB Statement

The operator SUB may be used interchangeably with the term SUBROUTINE. The parameters which follow the operator must be in the following sequence:

a. Subroutine Name - Any unique name identifying a user subroutine; follows general system-naming conventions.

b. Function - May be input or output, specifying the use of the subroutine for either input conversion or output conversion. If this is a subroutine for a Secondary Indexing function, CONVERT or ANALYZE must appear instead of INPUT or OUTPUT. If the subroutine is to be used for ANALYZE functions, the following length and mode entries are not needed. For all other functions (input, output, and convert), the entries must be present.

c. Input Length - Decimal integer, specifying the length of the input data field to be processed by the subroutine.

d. Output Length - Decimal integer, specifying the length of the data string which will be output by the subroutine.

e. Input Mode - May be ALPHA, DECIMAL, COORD, or BINARY indicating alphameric (EBCDIC), numeric, coordinate, or binary modes of operation. This parameter identifies the mode of the input to the subroutine, regardless of the function for which the subroutine is used. See subsection 2.3.2 of this volume.

f. Output Mode - Same as previous paragraph, where the mode specified applies to the form of output from the subroutine, regardless of the function for which the subroutine is used.

Examples:

```
SUB SUBNAME INPUT 7  7 ALPHA ALPHA.
```

This example indicates a subroutine named SUBNAME which is to be used for input data conversion. The length of the argument furnished to the subroutine is to be seven characters and the output function provided by the

subroutine is seven characters. The mode of the input to the subroutine is alphameric as is its output mode.

SUBROUTINE CONVERT OUTPUT 4 21 COORD ALPHA.

This example indicates a subroutine named CONVERT which is used for output conversion. It expects an input argument of four characters (bytes) and will provide an output function of 21 alphameric characters. The mode of the input argument is the special system mode of coordinates, discussed under the FIELD statement.

SUB SUB01 CONVERT 17 2 ALPHA DECIMAL

This example indicates a subroutine named SUB01 to be used to convert data in an index field from its file format to the format it will follow in the Index Data Set associated with the data file. It expects an input argument of 17 characters, alphameric mode, and will provide an output of two decimal characters.

SUBROUTINE SUB02 ANALYZE

This example indicates a subroutine named SUB02 to be used to analyze the parameter list of the FUNCTION operator to determine index usage and to provide a list of values for index qualification.

2.4.3.2.1 SUB Statement for Keyword Usage

The SUB statement, when used to define subroutines or tables applicable to keyword indexing, has the following parameters following the operator:

a. Subroutine Name - Any unique name identifying a user subroutine; follows general system-naming conventions.

b. Function - Identifies the function of the specified subroutine.

STOP: - Indicates a table containing the words that are to be eliminated from any further keyword processing.

DICTIONARY - May also be specified as DICT. This parameter indicates a table containing all values that are acceptable as keywords for the fild.

SCAN - Indicates this user-provided subroutine, rather than the System Scan routine, is to be indexed to process and extract keyword candidates from the keyword field in the transaction record.

Examples:

```
TAB    STOPNAM    STOP
TAB    DICTNAM    DICTIONARY
SUB    SCANNAM    SCAN
```

2.4.3.3 TAB Statement

Use of this statement is the same as for the SUB statement. The term TABLE may also be used.

2.4.3.4 EDIT Statement

This operator identifies an edit mask provided by the user. The format of the statement is the operator followed by a unique name (within this file and follows the general system name rules) and the user-supplied edit mask. The mask is delineated by single quotation marks (5-8 punch), and may contain any valid edit mask characters as defined for the NIPS 1410 FPS system with the exception of the floating dollar sign option, the debit option, the minus sign left option, and the asterisk protection option. An example of this statement is:

```
EDIT EDNAME '0.'.
```

This statement identifies the indicated edit mask and gives it the user-name EDNAME. For coding rules concerning

creation of edit masks, see Volume I, Introduction to File Concepts.

2.4.3.5 CLASS Statement

The operator CLASS for this statement may be substituted by the term CLASSIFICATION. The classification label follows and is enclosed within single quotation marks. Classification labels may not exceed 32 characters.

Example:

CLASSIFICATION 'TOP SECRET'.

2.4.3.5 FIELD Statement

The following parameters are used with the FIELD operator. Where sequence is critical, appropriate notation is made. When the presence or absence of the operand is optional, the word "Optional" appears in parenthesis in the heading for the description.

a. Field Name - Immediately follows the operator; must be a unique name (for the file) that follows the general system name rules: the length cannot exceed seven characters.

b. Field Length - Refers to the EBCDIC length of the data field. Where NUMER mode is specified, the actual field length may differ from the specified field length; however, this adjustment is internal to the system and does not affect the specification in the FIELD statement. Typically, this length is the length expected in the input transaction or the length to be produced by the system for output purposes. The entry is coded as a numeric integer, not to exceed three characters in length. High-order zeros may or may not be included.

c. Set/Function Identifier - Indicates in which set the field is to be structured. May be the single

alphabetic character X or C, indicating fixed field or record control field respectively, or may be a numeric integer not over three characters in length or greater than 255 in value, specifying the periodic set to which the field is to be assigned. May also contain a single alphabetic character prefix to indicate the following capabilities:

- "Cn", where "n" is any integer not greater than 255. Specifies a subset control field for the "nth" periodic set. This capability is to be used when the analyst desires the subset control field to be a data field of the set, rather than using the system-supplied "PSSQ" number for subset control.

- "Vn", where "n" is subject to the rules specified above. This form indicates the analyst desires the field to be defined as a variable field. "n" again specifies the periodic set in which this field is to be structured.

d. Field Mode Identifier (Optional) - Sequence of this and all following FIELD statement operands is not critical. Mode specification is optional. If not provided, the system will assume ALPHA mode, and issue an advisory message. The allowable modes are ALPHA, NUMER, and COORD. ALPHA indicates that the data field content is to be alphameric (EBCDIC) information and that arithmetic operations against this field will not be performed. NUMER indicates that the data field content is to be considered numeric data. COORD specifies the special system form for geographic coordinates. This data is maintained by the system in internal format for processing by geographic retrieval techniques and should be used only when such usage is expected. Note that automatic conversion of input and output is provided by the system when this mode is specified. Allowable input formats to this field are latitude and longitude as a single field or as separate fields defined as a group. Output formats

produced by the system conversion routines are also provided. Note that field lengths of 5, 6, 7, 8, 11, and 15 characters are permissible when this mode is specified. All others will cause error indications and the structuring process to be voided.

e. Input Subroutine Specification (Optional) - Subroutine or table name required for automatic conversion of this data entry should be included in the operand string. Note that this name must be included in a subroutine or table statement, as previously discussed in sections referring to SUB statements and TAB statements. The function (input or output) specified in the subroutine or table statement is assumed and cannot be overridden.

f. Output Subroutine Specification (Optional) -- Same as input subroutine specification above.

g. Edit Mask Specification (Optional) - Follows the same rule as the subroutine specification. The name provided for the edit mask must be defined on an edit statement, as previously discussed in the section under EDIT statement; this will cause the mask to be associated with the field for output editing. Note that both output subroutine processing and editing may be specified.

h. Output Title/Label (Optional) - The user may include an output title or label to be associated with the field when it is output under the QUIP component. Failure to include this operand will cause the QUIP component to use the field name for a title. When the label is to be included in the FFT, it is specified by delineating the desired "literal" with single quotation marks (5-8 punch).

Examples of FIELD statements:

```
FIELD CFLDONE 5 X  ALPHA  INSUB  OUTSUB  EDIT1
'CONTROL FIELD ONE'.
```

This example indicates that the field CFLDONE is to be five characters in length and a part of the fixed set to contain alphameric data, to be processed on input by the SUB named INSUB (assumes the existence of SUB statement), to be processed on output by the subroutine OUTSUB (same assumption), to be edited on output by the edit mask specified by the name EDIT1, and to include the output title or label CONTROL FIELD ONE. Note that the sequence of operands after the set identifier (X) is not critical.

FIELD MAJEQPT 7 3 ALPHA.

This example identifies a request for a field named MAJEQPT which is to be seven characters long, to be contained in the third periodic set, and to be alphameric information. No conversion or editing has been requested, and no output label was provided. Note that the sequence of all of the terms in this example is critical. Actually, ALPHA is excepted from this statement and, if used, the mode must not precede the set identifier.

2.4.3.7 GROUP Statement

The format of the GROUP statement follows a pattern similar to that established by the FIELD statement. Following the GROUP operator, the user-supplied name must be found. Standard system naming rules apply to the group name. The names of each of the fields/groups which are to be included in this group definition must follow the group name. Beyond this point in the operand string, the GROUP statement's format and sequence rules are exactly the same as those for the FIELD statement beginning with the rules for mode identification of the FIELD statement description.

2.4.3.8 INDEX Statement

The first entry must be INDEX. Following this, the user-supplied name must appear. This name must be defined as a field or group by an appropriate control statement. Standard system naming rules, of course, apply. Note: No

field or group longer than 30 bytes may be specified as an INDEX.

The following parameters apply to the INDEX statement. All operands are optional. Default parameters are noted.

a. Index Operand - ADD or DELETE; default is ADD. DELETE is used to delete an existing index during a UTNDX3PC run. It is of no consequence during FS.

b. File to Index Conversion Subroutine Specification (Optional) - The subroutine or table that will be invoked for automatic conversion from data file format to Index Data Set format. This name must be specified as a conversion routine in a SUB or TAB statement as described in section 2.4.3.2. If no subroutine is specified, values will be carried in their internal data file format.

c. Index Analysis Subroutine Specification (Optional) - The subroutine required to analyze and determine the qualification values for any FUNCTION operator in which the indexed field or group is a parameter. This name must be specified as an analyzer subroutine in a SUB or TAB statement as described in section 2.4.3.2. If no subroutine is specified, when the field is used in a FUNCTION operator clause, the clause will be bypassed by Index Processing.

Example of the INDEX statement:

```
INDEX CNTRY ADD
```

This example indicates that the field CNTRY will be designated as a file index. No subroutine will be used to convert data from the file format to a separate index format. Further, no analysis of a FUNCTION operator parameter list in which this field appears will be performed by Index Processing.

2.4.3.8.1 INDEX Statement for Keyword Fields

The first entry must be INDEX. Following this the user-supplied name must appear. This name must be defined as a field or variable set by the appropriate control statement. Standard system naming rules apply. A field may not be defined as both a keyword and secondary index.

The following parameters apply to the INDEX statement.

a. Index Operand - (ADD or DELETE) - Default is ADD. DELETE is used to eliminate an existing index and is used only by the index specification function performed by the utility UTNDXSPC.

b. KEYWORD - Designation of this statement as one that defines a keyword field rather than the standard secondary index field. This parameter is mandatory for an ADD operation, but optional for a DELETE operation as the field has already been designated as a keyword field.

c. Hyphen Designation - The user can control processing of the hyphen by the system scan routine. A hyphen is always treated as a word separator if it follows a word separator character. If it is followed by a text character, one of four options can be applied. TEXT is the default option which means the hyphen is to be treated and retained as part of the value. The other options are DROP, RETAIN and SEPARATE.

DROP - The hyphen is either superfluous or it is a word separator, depending upon its context with respect to adjacent characters. It is superfluous if: (1) text characters immediately precede and follow it, or (2) it is immediately preceded by a text character. In this case, blanks are considered to be superfluous. If neither of these cases apply, the hyphen is treated as a word

separator. In any case, the hyphen is dropped.

RETAIN - The hyphen is subject to the same rules as the DROP option, except that in the two cases where it is considered superfluous it is retained in the value as a text character.

SEPARATE- The hyphen is always treated as a word separator.

d. Stop Table - (Optional) This is the name of the user provided table that is to be applied against the values in this field. If a value is found in the table, it is to be eliminated from any further consideration as a keyword. If a value is not present in the table, or if no table were specified, the value becomes a keyword candidate. This table must have been identified by a SUB/TAB statement and have been defined as a STOP table.

e. Dictionary - (Optional) This is the name of the user provided table containing all nonliteral values from this field that may become keywords. This table must have been identified by a SUB/TAB statement and have been defined as a DICTIONARY. If no dictionary is specified, all values in the field, unless some were eliminated by a Stop Word Table, become keywords. Literal words always become keywords.

f. Scan Subroutine - (Optional) This is the name of the user provided subroutine that is to be invoked to process the keyword field. This subroutine must have been identified as a SCAN subroutine. If no subroutine is specified, the system's scan routine will select the keyword candidates from the field.

Example:

INDEX VSET1 ADD KEYWORD TAB1 TAB2.

This example indicates the field name VSET1, which has already been defined as a variable set, is to be a keyword index. When processing this field in the transaction record, TAB1, defined as a stop word table and TAB2, defined as a dictionary are to be used to determine what values are to become keywords.

2.4.3.9 VSET Statement

The VSET statement defines a variable set for the user's data file. More than one variable set may be defined with a VSET statement supplied for each one.

The operator for the statement is the term VSET which appears first. The following parameters are used:

a. Variable Set Name - Must immediately follow the operator and is unique for the file.

b. Variable Set Display Size - Decimal number stating the width of the variable set to be printed per line during QUIP processing.

c. Variable Set Label (Optional) - Label of up to 69 characters in length may be associated with the variable set, and must be enclosed within single quote marks.

2.4.3.10 NOTE Statement

This statement provides the capability to have explanatory comments about the source deck recorded on the printout produced by FS. The NOTE operator must occur first in the statement. The information following will be printed on the source printout produced by FS. No periods may be included within the body of the NOTE statement, since such an occurrence would signify the end of the note. Other than

this restriction, all other valid EBCDIC characters may be used. The NOTE statement may occur anywhere in the input stream after the structure statement and is not subject to the sequencing rules already specified or those discussed in the following section.

2.4.3.11 END Statement

The last statement in the FS source deck uses the single term END or ENDFS to signify the last user-supplied statement.

2.5 File Structuring for a Non-NIPS File

FS can be used to create an FFT which describes all the usable data elements of a non-NIPS data file. This description includes the definition of various record formats, record type fields used to identify the different record formats, record ID fields, and FILLER fields to account for elements within a record format to be ignored.

2.5.1 General Considerations

In describing a non-NIPS file via FS, the file must conform to various constraints and restrictions. These deal with general file organization, logical record size, record type fields, record ID fields, complete format description and data content.

2.5.1.1 File Organization

To be described by FS, a non-NIPS file must contain either fixed length or variable length records. Fixed length records can be up to 996 bytes in length. Variable length records can be up to 1,000 bytes, but the first four bytes must contain the length of the record. The records in the file may be either blocked or unblocked.

2.5.1.2 Record Formats

The records within a non-NIPS file may have many different formats. Up to 256 different formats may be contained with a single file. Each format must be identified by a single record type field. This field must be present in all records, have the same location in all records, and have the same length in all records (up to 10 bytes).

Of the possible 256 different record formats, one must be identified as a nonrepeating (fixed) format. This format will be treated like the fixed set in a NIPS record and it must be present for each unique record ID. Only one occurrence of the nonrepeating format is permitted within a record ID. All other formats are considered to be repeating formats and will be treated as periodic sets in a NIPS record. Their presence is optional within a record ID.

When describing the format of a record type, each byte in the record must be included in a field definition. These include data fields which are not relevant to query requirements and various system fields such as the record length field. Fields which are to be ignored can be designated by the special, repeatable field name FILLER. FILLER fields cannot be referenced beyond definition.

2.5.1.3 Record ID Fields

Each record must have record identification data which is contained in one or more user designated fields. These fields need not be contiguous; but each field must be present in each record. The field positions may differ between record types, but the field size and its relative position in the record ID must not differ. The length of the record ID cannot exceed 256 bytes.

2.5.1.4 Data Field Restrictions

Data fields will be processed as either alphabetic, numeric, or coordinate; however, numeric and coordinate mode

fields must adhere to certain restrictions. To be processed as numeric, a data field must be either a binary fullword or a zoned decimal field. All other numeric data fields (non-fullword binary, floating point, and packed decimal) must be defined as alpha and have an associated subroutine to provide output translation.

Data fields which are defined as coordinate must conform to the standard NIPS coordinate data field format. The data must be in binary word format, a fullword each for latitude and longitude, and must be translatable by the NIPS coordinate subroutines into a standard external 11 or 15 character geographic coordinate. Otherwise, the field must be defined as alpha.

A single variable length data field may be defined for each record type provided the field is the last field in the record type. Also the record type must include a fullword binary field containing the length of the variable field.

2.5.2 Language Statements

2.5.2.1 STRUCTURE Statement

The STRUCTURE statement, in addition to defining the file name, will initiate creation of a non-NIPS FFT, when the keyword NONIPS follows the file name. The file name must still conform to all standard NIPS conventions.

Example:

```
STRUCTURE NNTEST NONNIPS.
```

2.5.2.2 DEFINE Statement

The DEFINE is used to define the beginning of a new record type and must precede all FIELD/GROUP statements for the record type. The parameters which follow the operator must be in the following sequence:

a. Record type sequence number - Either an "Y" for the first record type defined or a one-up sequence number for each additional record type (1 through 255). An "X" denotes that the record type is non-repeating. The nonrepeating record type must be the first record type defined.

b. Record type code - The value with a record which will uniquely identify the record type. The code may be from 1 through 10 characters in length and may be expressed in either character or hexadecimal representation. For character representation, the code is entered as alphanumeric characters, enclosed in single quotes if any special characters are used. Embedded single quotes are not permitted. For hexadecimal representation, the code is entered in hexadecimal notation, enclosed in single quotes, and preceded with an X. The code, in hexadecimal, may represent a maximum of 10 bytes.

Example using character representation:

```
DEFINE X ABC9.
DEFINE 1 '71*Z'.
```

Example using hexadecimal representation:

```
DEFINE 2 X'F1E4FF40'.
```

NOTE: The DEFINE statement is not permitted when structuring a standard NIPS FFT.

2.5.2.3 FIELD Statement

The FIELD statement is used to describe each field in the non-NIPS file. The following parameters are used following the FIELD operator:

a. Field Name (required) - Immediately follows the operator. Any name conforming to the standard NIPS

naming conventions can be used with the following restrictions:

1. FILLER is the name used to account for unneeded or unused bytes within a record type.

2. VSZn is the name used to reference a binary fullword field which contains the length value for the variable field in the record type. In the name VSZn, "n" is omitted when referenced in the nonrepeating record type and "n" is the record type sequence number when referenced in a repeating record type.

3. When the name is applied to a record ID field, the name must be duplicated for the redefinition of the field in each record type. When the name is applied to the record type field, the name maybe duplicated for each record type. In this case, any reference in QUIP to the field name will apply to the fixed record type only.

b. Field Length (required) - The length of the field with the following considerations:

1. Record ID fields - The length must be the same for all occurrences of the same record ID field name.

2. Record type field - The length can be a maximum of ten bytes and must remain constant for each redefinition of the field. The length must be sufficient to contain the record type code entered on the preceding DEFINE statement.

3. NUMER fields - The length is the number of positions to be output. The internal length is assumed to be four bytes an a full word boundary unless the field is contained in a group; then the internal length is the same as the output length.

4. COORD fields - The output length of the coordinate field. Acceptable lengths are 5,6,7,8,11 and 15. The internal length is assumed to be four bytes if the output length is less than 11 or eight bytes if the output length is greater than 8. Internal, the field is assumed to be on a full word boundary.

5. ALPHA, DECML and FILLER fields - The length is the number of bytes contained in the field in the file.

c. Set/Function Identifiers (required) - The set sequence number from the preseeding DEFINE statement or one of the following codes for special functional use:

1. "Cn" is used to identify the field as a portion of the record ID. The "n" is a 1- to 3-digit number denoting the fields relative position in the ID. When used on a redefinition of a record ID field, the "Cn" value must be the same as on the original definition.

2. "T" is used to identify the field as the record type field. This field will contain the record type code as specified on the preceding DEFINE statement.

3. "Vn" is used to identify the field as a variable field. The "n" is the set sequence number from the preceding DEFINE statement. A field identified as a variable field must have an associated "VSZn" field and must be the last field defined in the record type description.

d. Field Mode Identifier (Optional) - If specified, this is the mode of the field. Valid entries are:

1. ALPHA, used to identify the field as containing either alphanumeric data or

nonstandard numeric data. Nonstandard numeric data includes packed decimal fields and binary fields which are not fullwords. No arithmetic operations are permitted against ALPHA fields. ALPHA is the only valid mode for record ID and record type fields.

2. NUMER, used to identify the fields as containing standard numeric data. The data will be assumed to be fullword binary unless the field is contained in a group, in which case it will be assumed to be zoned decimal (EBCDIC).

3. DECML, used to identify the field as containing zoned decimal numeric data. This mode is not permitted when executing a standard NIPS File Structure.

4. COORD, used to identify the field as containing coordinate data in the standard internal NIPS binary word format.

e. Other Operands - Additional operands which may be specified are:

1. Input Subroutine Specification

2. Output Subroutine Specification

3. Edit Mask Specification

4. Output Title/Label

The uses of these operands is the same as when specified for a standard NIPS FS and are described in section 2.4.3.6.

2.5.2.4 GROUP Statement

The GROUP statement is used to identify a sequence of adjacent field/groups to be collectively referenced via a new name. The rules which apply to defining a group for standard NIPS files also apply for non-NIPS files as described in sections 2.3.6 and 2.4.3.7.

When defining a group for a non-NIPS file, the group cannot contain any FILLER fields, nor can it be named FILLER. Additionally, no fullword binary fields can be included in a group.

2.5.2.5 Other FS Statements

The INDEX and VSET statements are not allowed for non-NIPS files. The SUB, TAB, EDIT, CLASS, NOTE and END statements are used in the same manner as for standard NIPS files.

Section 3

OUTPUT

Output from the file structuring program consists of a listing of the input deck with messages noting any diagnosed errors. Some messages are only advisory, but most of them relate to errors which prevent the structuring of the FFT. If the FFT is successfully structured, it is printed out immediately following the deck listing and the individual records which make up the FFT are written on the direct access device saved for file generation. The newly structured FFT can be contained in an Indexed Sequential Access Method (ISAM) or a Virtual Storage Access Method (VSAM) data set as detailed in Volume VIII, Job Preparation Manual. At that time the actual data records are created and will reside with the FFT on the same volume. During file structuring, the user has the option of specifying a file block size greater than 1,004. If he does not specify a block size, file structuring will use the standard 1,004 size. Details for block size specifications can be found in Volume VIII, Job Preparation Manual.

3.1 Error and Exception Detection

Some errors are abortive in constructing an FFT, while others are merely advisory in nature. All errors are listed on the system output device as they are discovered in each input card submitted. At the end of a job, the FFT is constructed if no abortive errors were found during the processing of the input deck.

3.2 Operation for the File Structure Component

This subsection contains a brief flowchart of the operations performed by FFTs during execution.

FILE STRUCTURING (FS)

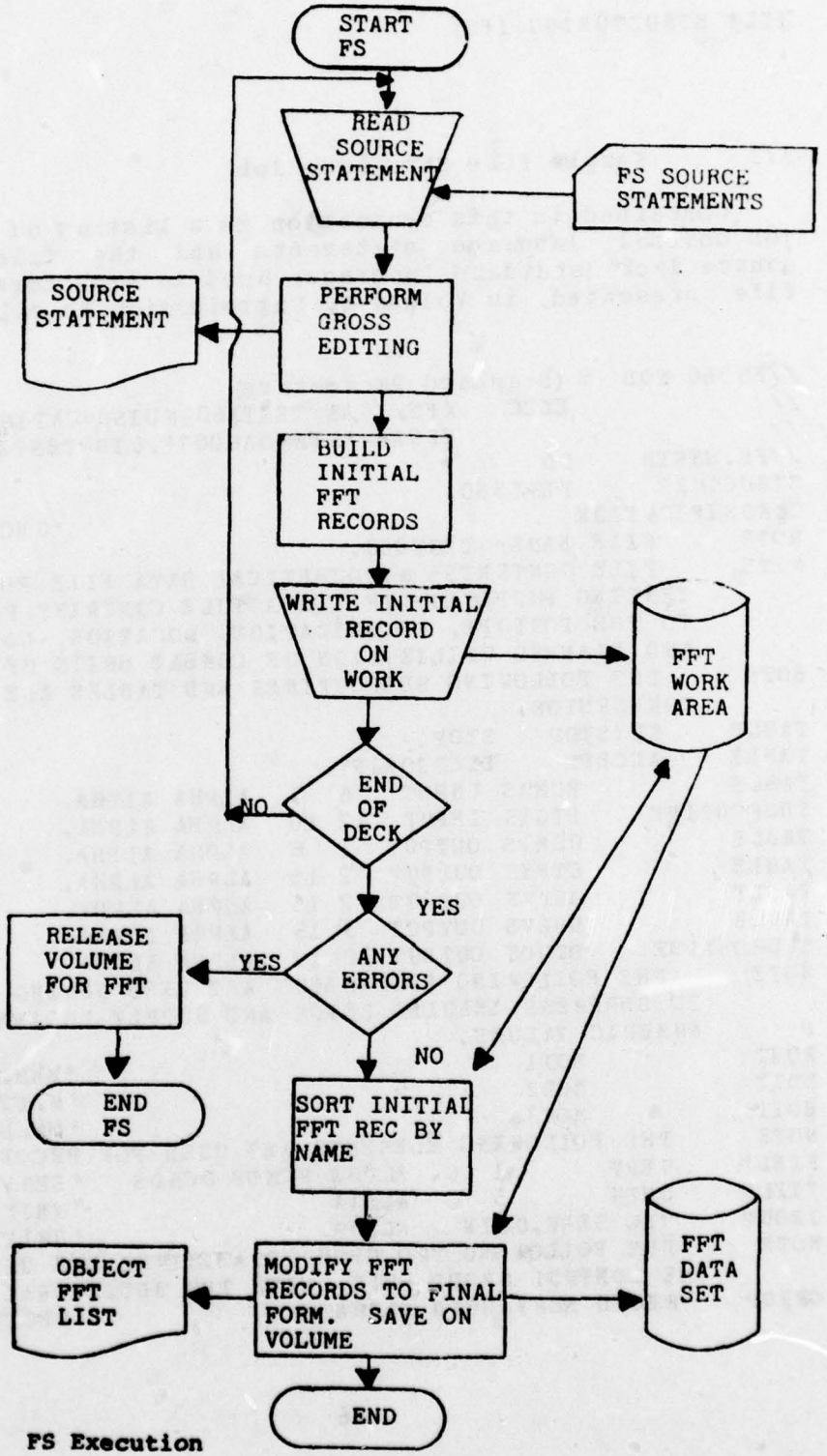


Figure 2. FS Execution

3.3 Sample File Structure Job

Contained in this subsection is a listing of both the OS job control language statements and the file structure source deck (standard language) used to structure the sample file presented in Volume I, Introduction to File Concepts.

```
//FS360 JOB    (Standard Parameters)
//            EXEC   XFS,ISAM=TEST360,NDISP=CATLG,
//                   VISAM='SER=DA0001',LIB=TEST360
//FS.SYSIN    DD     *
STRUCTURE     TEST360.
CLASSIFICATION                                  'UNCLASSIFIED'.
NOTE      FILE NAME- TEST360.
NOTE      FILE CONTENTS- HYPOTHETICAL DATA FILE FOR TRAINING AND
        TESTING PURPOSES- THE DATA FILE CONTAINS PSEUDO DATA RELATING
        TO THE POSTURE, ORGANIZATION, LOCATION, COMMAND, EQUIPMENT,
        AND PLANNED UTILIZATION OF COMBAT UNITS OF THE ARMED FORCES.
NOTE      THE FOLLOWING SUBROUTINES AND TABLES ARE USED FOR DATA
        CONVERSION.
TABLE     KEYSTOP    STOP.
TABLE     ACCEPT     DICTIONARY.
TABLE          RCMDS INPUT   6  1   ALPHA ALPHA.
SUBROUTINE     DTGIS INPUT  12 10   ALPHA ALPHA.
TABLE          OCMDS OUTPUT  1  6   ALPHA ALPHA.
TABLE          CTRYS OUTPUT  2 15   ALPHA ALPHA.
TABLE          ACTVS OUTPUT  2 15   ALPHA ALPHA.
TABLE          UNLVS OUTPUT  3 15   ALPHA ALPHA.
SUBROUTINE     DTGOS OUTPUT 10 12   ALPHA ALPHA.
NOTE      THE FOLLOWING EDIT MASKS ARE USED DURING OUTPUT PROCESSING
        TO SUPPRESS LEADING ZEROS AND SUPPLY DECIMAL POINTS FOR
        NUMERIC VALUES.
EDIT           MOD1                             '##0'.
EDIT           MOD2                             '#.##'.
EDIT           MOD3                             '##.#'.
NOTE      THE FOLLOWING ELEMENTS ARE USED FOR RECORD CONTROL.
FIELD     SERV       1  C  ALPHA RCMDS OCMDS   'SERVICE BRANCH'.
FIELD     UUIN       5  C  ALPHA               'UNIT IDENTIFIER'.
GROUP     UIC SERV,UUIN    ALPHA               'UNIT ID CODE'.
NOTE      THE FOLLOWING TWO GROUP STATEMENTS ARE USED TO REDEFINE
        THE CONTROL GROUP -UIC- WITH TWO ADDITIONAL NAMES.
GROUP     RECID SERV,UUIN ALPHA                'RECORD CONTROL'.
```

FILE STRUCTURING (FS)

```
GROUP    CONTROL SERV,UUIN ALPHA                    'RECORD CONTROL'.
NOTE     THE FOLLOWING ELEMENTS ARE CONTAINED IN THE FIXED SET OF
       THE DATA RECORD.
FIELD    UNTYY      4  X  ALPHA                     'UNIT TYPE CODE'.
FIELD    UNTYZ      1  X  ALPHA                     'MAJOR UNIT INDICATOR'.
FIELD    UNLVL      3  X  ALPHA      UNLVS          'MAJOR ORG LEVEL'.
GROUP    UNTLV      UNTYY,UNTYZ,UNLVL               'UNIT TYPE AND LEVEL'.
FIELD    HOME       1  X  ALPHA   RCMDS DCMDS       'HOME COMMAND'.
FIELD    UNFLG      1  X  ALPHA                     'UNIT FLAG'.
FIELD    MJFOR      1  X  ALPHA                     'MAJOR FORCE IND.'.
FIELD    PREV       1  X  ALPHA   RCMDS DCMDS       'PREVIOUS COMMAND'.
FIELD    ATACH      1  X  ALPHA   RCMDS DCMDS       'ATTACHED COMMAND'.
FIELD    FUTU       1  X  ALPHA   RCMDS DCMDS       'FUTURE COMMAND'.
FIELD    TRDTG     10  X  ALPHA   DTGIS DTGOS       'TRANSFER DATE'.
FIELD    UNRDY      2  X  ALPHA                     'READINESS STATUS'.
FIELD    REASN      1  X  ALPHA                     'REASON'.
FIELD    RATTN      2  X  ALPHA                     'ATTAINABLE STATUS'.
GROUP    RECDE UNRDY,REASN,RATTN                    'READINESS GROUP'.
FIELD    RADTG     10  X  ALPHA   DTGIS DTGOS       'ATTAIN. STATUS DTG'.
FIELD    UNIT      12  X  ALPHA                     'UNIT NAME-SHORT'.
FIELD    UNAME     27  X  ALPHA                     'UNIT NAME-FULL'.
FIELD    OPCON      6  X  ALPHA                     'OP CONTROL UNIT'.
FIELD    COMDR     20  X  ALPHA                     'CO NAME AND RANK'.
FIELD    LOC       18  X  ALPHA                     'LOCATION NAME'.
FIELD    POINT     11  X  COORD                     'HQ COORD LOCATION'.
FIELD    DAPT1     11  X  COORD                     'DUTY AREA POINT 1'.
FIELD    DAPT2     11  X  COORD                     'DUTY AREA POINT 2'.
FIELD    DAPT3     11  X  COORD                     'DUTY AREA POINT 3'.
FIELD    DAPT4     11  X  COORD                     'DUTY AREA POINT 4'.
GROUP    AREA DAPT1,DAPT2,DAPT3,DAPT4 COORD         'GEO-DUTY AREA'.
FIELD    CNTRY      2  X  ALPHA   CTRYS            'COUNTRY CODE'.
FIELD    CNAM      15  X  ALPHA                     'COUNTRY NAME'.
FIELD    GEPOL      2  X  ALPHA   CTRYS            'GEOPOLITICAL AREA'.
FIELD    PERS       6  X  NUMER   MOD1             'AUTHORIZED PERSNL'.
FIELD    ACTIV      2  X  ALPHA   ACTVS            'CURRENT ACTIVITY'.
FIELD    LAUD      10  X  ALPHA                     'DTG LAST UPDATE'.
FIELD    LYN        1  X  ALPHA                     'LOCATION STATUS'.
FIELD    RPERS      1  X  NUMER                     'PERSNL READINESS'.
FIELD    RSPLY      1  X  NUMER                     'SUPPLY READINESS'.
FIELD    REQPT      1  X  NUMER                     'EQUIPMENT READINESS'.
FIELD    RTRNG      1  X  NUMER                     'TRAINING READINESS'.
GROUP    RMGRP RPERS,RSPLY,REQPT,RTRNG NUMER        'READY GROUP'.
FIELD    READAVG    3  X  NUMER   MOD2             'READINESS AVERAGE'.
```

FILE STRUCTURING (FS)

```
FIELD     RITNM      3  X   NUMER   MOD3          'RADIUS-NAT. MILES'.
FIELD     UNTYP      5  X   ALPHA                 'UNIT TYPE CODE'.
FIELD     TPNAM     42  X   ALPHA                 'UNIT TYPE NAME'.
FIELD     UNTOE     17  X   ALPHA                 'T/O AND E REF.'.
FIELD     HIER      11  X   ALPHA                 'HIERARCHY CODE'.
FIELD     COMMENT   50  VX  ALPHA                 'COMMENT'.
NOTE THE FOLLOWING ELEMENTS ARE USED FOR PERIODIC SET 1 CONTROL.
FIELD     MECL       3  C1  ALPHA                 'MAJOR EQP. CLASS'.
FIELD     MEQPT     10  C1  ALPHA                 'MAJOR EQP. ID'.
GROUP     MECLQ MECL,MEQPT                        'EQP. CLASS AND TYPE'.
NOTE   THE FOLLOWING ELEMENTS ARE CONTAINED IN PERIODIC SET 1 OF THE
     DATA RECORD.
FIELD     MEMOD     10  1   ALPHA                 'MAJOR EQP. MODEL'.
FIELD     MENAM     18  1   ALPHA                 'MAJOR EQP. NAME'.
FIELD     MEPCAP     1  1   ALPHA                 'WEAPON DEL. TYPE'.
FIELD     MEPSD      3  1   NUMER                 'NUMBER POSSESSED'.
FIELD     MEADA      3  1   NUMER                 'NUMBER ON ALERT'.
FIELD     MEORC      3  1   NUMER                 'NO. CONVEN. OP READY'.
FIELD     MEORN      3  1   NUMER                 'NO. NUCLEAR OP READY'.
FIELD     MESQP      3  1   NUMER                 'NO. SPEC. ALERT EQPS'.
FIELD     MESWP      3  1   NUMER                 'NO. SPEC. ALERT WPNS'.
GROUP     MESIA  MESQP,MESWP NUMER                'SPEC. ALERT TALLY'.
FIELD     MESIC      3  1   NUMER                 'NO. SPEC. COMMITTED'.
FIELD     MEREC     10  1   ALPHA                 'RECONN. CAPABILITY'.
FIELD     MEDEP      1  1   ALPHA                 'DEPLOYMENT ID'.
FIELD     MEDDT      5  1   NUMER                 'DEPLOYMENT DATE'.
FIELD     MEDUR      1  1   ALPHA                 'DEPLOYMENT TIME CODE'.
FIELD     MELYN      1  1   ALPHA                 'DEPLOY. LOC. STATUS'.
FIELD     MELOC     18  1   ALPHA                 'DEPLOYMENT LOCATION'.
FIELD     MEPNT     11  1   COORD                 'DEPLOYMENT COORD'.
FIELD     METRY      2  1   ALPHA   CTRYS         'DEPLOY. COUNTRY CODE'.
FIELD     MEPOL      2  1   ALPHA   CTRYS         'DEPLOY. GEOPOLITICAL'.
FIELD     MECNA     15  1   ALPHA                 'DEPLOYMENTS COUNTRY'.
NOTE   THE FOLLOWING ELEMENTS ARE CONTAINED IN PERIODIC SET 2 OF THE
     DATA RECORD.
FIELD     SECLASS    3  2   ALPHA                 'SECONDARY EQP CLASS'.
FIELD     SEMODEL   10  2   ALPHA                 'SECONDARY EQP MODEL'.
FIELD     SENAME    18  2   ALPHA                 'SECONDARY EQP NAME'.
FIELD     SEPOSSD    4  2   NUMER                 'SECONDARY EQP POSS.'.
FIELD     SEAUTH     4  2   NUMER                 'SECONDARY EQP AUTH.'.
NOTE   THE FOLLOWING ELEMENTS ARE CONTAINED IN PERIODIC SET 3 OF THE
     DATA RECORD.
FIELD     PLAN       4  3   NUMER                 'PLAN ID'.
```

FILE STRUCTURING (FS)

```
FIELD     PLEAC      1  3  ALPHA                    'PLAN STATUS'.
FIELD     PLDTG     10  3  ALPHA  DTGIS DTGOS       'PLAN STATUS'.
FIELD     PLPST      1  3  ALPHA                    'DTG AVAILABLE'.
FIELD     PLFDG      1  3  ALPHA                    'EXPECTED PLAN STATUS'.
FIELD     PLRT       6  3  ALPHA                    'PLAN RESPONSE TIME'.
FIELD     PLTRT      6  3  ALPHA                    'STAGING TIME'.
NOTE   THE FOLLOWING ELEMENT IS CONTAINED IN PERIODIC SET 4 OF THE
    DATA RECORD.
FIELD     TRTY       6  4  ALPHA                    'TREATY NAME'.
NOTE   THE FOLLOWING ELEMENTS ARE CONTAINED IN PERIODIC SET 5 OF THE
    DATA RECORD.
FIELD     NAME      18  5  ALPHA                    'NAME'.
FIELD     RANK       4  5  ALPHA                    'RANK'.
FIELD     SERNUMR    6  5  ALPHA                    'SERIAL NUMBER'.
FIELD     SERVICE    1  5  ALPHA                    'SERVICE'.
FIELD     ASSGN     20  5  ALPHA                    'DUTY ASSIGNMENT'.
FIELD     SPCODE     5  5  ALPHA                    'SPECIALITY CODE'.
NOTE   THE FOLLOWING ELEMENTS ARE CONTAINED IN PERIODIC SET 6 OF THE
    DATA RECORD.
FIELD     SBUIC      6  6  ALPHA                    'SUBORDINATE UNIT UIC'.
FIELD     SBFLG      6  6  ALPHA                    'SUBORD. UNIT FLAG'.
NOTE   THE FOLLOWING ELEMENT IS CONTAINED IN A VARIABLE SET OF THE
    DATA RECORD.
INDEX     COMMENT    ADD   KEYWORD.
INDEX     REFER      ADD   KEYWORD   KEYSTOP   ACCEPT.
VSET      REFER     50                          'UNIT REMARKS'.
INDEX     SERV   ADD.
INDEX     CNTRY  ADD.
INDEX     MEDDT  ADD.
INDEX     METRY  ADD.
INDEX     PLRT   ADD.
INDEX     SPCODE ADD.
END.
/*
```

3.4 Sample Non-NIPS FS Job

Contained in this subsection is a listing of both the OS job control language statements and the file structure source deck used to structure a sample non-NIPS FFT.

```
//FSNNJOB JOB    (Standard Parameters)
//  EXEC XFS,
//       ISAM='NN.TEST360.FFT',VISAM='SER=DA0001',
//       PRIME=1,NDISP=CATLG,LIB=TEST360
//FS.SYSIN DD *
STRUCTURE FFT NONNIPS.
NOTE -- THIS FFT IS DESIGNED FOR USE WITH A NON-NIPS VERSION OF THE
        TEST360 FILE SHOWN IN PART 3-3 OF THIS VOLUME.
CLASSIFICATION      UNCLASSIFIED.
NOTE -- EACH RECORD TYPE (SET) MUST BEGIN WITH A DEFINE STATEMENT.
NOTE -----DEFINE RECORD TYPE TO ACT AS FIXED SET.
DEFINE X 'A'.
NOTE ----- NEXT FIELD IS FOR SPACING - NOT ADDRESSABLE.
FIELD    FILLER    2    X    ALPHA
NOTE ----- DEFINE RECORD ID FIELDS.
FIELD    SERV      1    C1   ALPHA 'RECORD ID- PART 1'.
FIELD    UUIN      5    C2   ALPHA 'RECORD ID- PART 2'.
GROUP    UIC    SERV  UUIN  ALPHA 'RECORD ID- GROUP'.
NOTE ----- NEXT FIELD IS FOR SPACING - NOT ADDRESSABLE.
FIELD    FILLER    2    X    ALPHA.
NOTE ----- DEFINE RECORD TYPE FIELD.
FIELD    RTYPE     1    T    ALPHA.
FIELD    HOME      1    X    ALPHA.
FIELD    MJFOR     1    X    ALPHA.
FIELD    ATACH     1    X    ALPHA.
FIELD    TRDTG    10    X    ALPHA.
FIELD    UNRDY     2    X    ALPHA.
FIELD    REASN     1    X    ALPHA.
FIELD    RATTN     2    X    ALPHA.
GROUP    RECDE UNRDY REASN RATTN ALPHA.
NOTE ----- NEXT FIELD IS DECIMAL (DECML).
FIELD    YRDTG     2    X    DECML.
FIELD    DADTG     3    X    DECML.
GROUP    TIDTG  YRDTG DADTG.
FIELD    HRDTG     4    X    DECML.
FIELD    TZDTG     1    X    ALPHA.
GROUP    RADTG TIDTG HRDTG TZDTG.
```

FILE STRUCTURING (FS)

```
FIELD     UNIT      12     X    ALPHA.
FIELD     UNAME     27     X    ALPHA.
FIELD     OPCON      6     X    ALPHA.
NOTE ----- DEFINE 1ST REPEATING RECORD FORMAT.
DEFINE 1 'B'.
NOTE ----- NEXT FIELD IS FOR SPACING - NOT ACCESSIBLE.
FIELD     FILLER     2     1    ALPHA.
NOTE ----- REDEFINE RECID ID FIELDS, BUT NOT THE GROUP.
FIELD     SERV       1     C1   ALPHA.
FIELD     UUIN       5     C2   ALPHA.
NOTE ----- SPACING.
FIELD     FILLER     2     1    ALPHA.
NOTE ----- REDEFINE RECORD TYPE FIELD.
FIELD     RTYPE      1     T    ALPHA.
FIELD     COMDR     20     1    ALPHA.
FIELD     LOC       18     1    ALPHA.
NOTE ----- NEXT FIELD CONTAINS COORDINATE DATA IN ALPHA FORMAT.
FIELD     POINT     11     1    ALPHA.
FIELD     CNTRY      2     1    ALPHA.
FIELD     CNAM      15     1    ALPHA.
FIELD     GEPOL      2     1    ALPHA.
NOTE ----- SPACING TO END OF RECORD FORMAT.
FIELD     FILLER     5     1    ALPHA.
NOTE ----- DEFINE 2ND REPEATING RECORD FORMAT.
DEFINE 2 'C'.
NOTE ----- SPACING.
FIELD     FILLER     2     2    ALPHA.
NOTE ----- REDEFINE RECORD ID.
FIELD     SERV       1     C1   ALPHA.
FIELD     UUIN       5     C2   ALPHA.
NOTE ----- SPACING.
FIELD     FILLER     2     2    ALPHA.
NOTE ----- REDEFINE RECORD TYPE
FIELD     RTYPE      1     T    ALPHA.
NOTE ----- SPACING TO FORCE ALIGNMENT OF NEXT FIELD.
FIELD     FILLER     1     2    ALPHA.
NOTE ----- PERS IS BINARY NUMERIC FIELD.
NOTE ----- FOUR BYTES ON FULLWORD BOUNDARY.
FIELD     PERS       6     2    NUMER.
FIELD     ACTIV      2     2    ALPHA.
FIELD     LAUD      10     2    ALPHA.
FIELD     LYN        1     2    ALPHA.
FIELD     UNTYP      5     2    ALPHA.
```

```
FIELD     TPNAM     42     2    ALPHA.
NOTE ----- DEFINE 3RD REPEATING RECORD FORMAT.
DEFINE 3 'F'.
NOTE ----- REDEFINE RECORD ID AND TYPE FIELDS.
FIELD     FILLER     2     3    ALPHA.
FIELD     SERV       1     C1   ALPHA.
FIELD     UUIN       5     C2   ALPHA.
FIELD     FILLER     2     3    ALPHA.
FIELD     RTYPE      1     T    ALPHA.
NOTE ----- DEFINE REST OF FORMAT.
FIELD     PLAN       4     3    DECML.
FIELD     PLFAC      1     3    ALPHA.
FIELD     PLDTG     10     3    ALPHA.
FIELD     PLFST      1     3    ALPHA.
FIELD     PLFDG     10     3    ALPHA.
FIELD     PLRT       6     3    ALPHA.
FIELD     PLTRT      6     3    ALPHA.
NOTE ----- DEFINE 4TH REPEATING RECORD FORMAT.
DEFINE 4 'G'.
NOTE ----- REDEFINE RECORD ID AND TYPE FIELDS.
FIELD     FILLER     2     4    ALPHA.
FIELD     SERV       1     C1   ALPHA.
FIELD     UUIN       5     C2   ALPHA.
FIELD     FILLER     2     4    ALPHA.
FIELD     RTYPE      1     T    ALPHA.
FIELD     PRTY       6     4    ALPHA.
NOTE ----- DEFINE 5TH REPEATING RECORD FORMAT.
DEFINE 5 'H'.
NOTE ----- REDEFINE RECORD ID AND TYPE FIELDS.
FIELD     FILLER     2     5    ALPHA.
FIELD     SERV       1     C1   ALPHA.
FIELD     UUIN       5     C2   ALPHA.
FIELD     FILLER     2     5    ALPHA.
FIELD     RTYPE      1     T    ALPHA.
FIELD     SUBUIC     6     5    ALPHA.
FIELD     SUBFLG     1     5    ALPHA.
NOTE ----- DEFINE 6TH REPEATING RECORD FORMAT.
DEFINE 6 'I'.
NOTE ----- REDEFINE RECORD ID AND TYPE FIELDS.
FIELD     FILLER     2     6    ALPHA.
FIELD     SERV       1     C1   ALPHA.
FIELD     UUIN       5     C2   ALPHA.
FIELD     FILLER     2     6    ALPHA.
```

FILE STRUCTURING (FS)

```
FIELD    RTYPE     1    T   ALPHA.
NOTE ----- SPACE FOR ALIGNMENT.
FIELD    FILLER    1    6   ALPHA.
NOTE ----- VSZ6 IS THE SIZE FIELD FOR THE VARIABLE FIELD
             IN THIS SET.
NOTE         IT MUST BE AN ALIGNED FULLWORD BINARY FIELD.
FIELD    VSZ6      4    6   NUMER.
NOTE ----- REFER IS A VARIABLE FIELD.
FIELD    REFER    50    V6  ALPHA.
END.
/*
```

Section 4

FILE REVISION

File Revision (FR) revises the format in which data is stored in an FFS data file. FR provides the restructuring of an existing data file, allowing the insertion or removal of fields or sets of data and also allowing for a new optional file block size.

File Revision will compare the FFT describing the file in its current format with the FFT describing the file in its revised format. FR will then generate FM logic statements which will be used by FM for copying selected data elements into the new format.

FR will process data on a field basis only. It will permit the addition, deletion, and limited relocation of fields as well as changes to their storage mode, size, and name. Storage mode (alphabetic, zoned decimal, binary, coordinate) changes are permitted with two exceptions. FR will not allow a binary or zoned decimal field to be changed to a coordinate field, nor will it allow a coordinate field to be changed to a binary or zoned decimal field. Field sizes must conform to File Structuring rules. Alphameric field size changes result in truncation or blank padding on the right. Zoned decimal field changes in truncation or zero fill on the left. Specification of numeric (binary) field size changes affect only the external representation of the field.

Periodic sets may be deleted or relocated. They may also be added, but no data will be included, as a direct result of FR, in either new sets or fields. FR will permit fields from any set in the old file to be split into several sets in the new file. It will not permit fields from multiple sets in the old file to be merged into one set in

the new file. There is no way for FR to logically connect the subsets of two existing sets.

FR will permit the addition of major or subset control fields. Major or subset control fields may be deleted as long as the retained control fields (i.e., the control fields common to both the old and new files) provide a unique key for each record. Control fields may be added and deleted within the same FR run if the retained control fields provide unique keys. If the retained control fields will not provide unique keys, two FR runs must be made: the first to add the additional fields, and the second to delete unwanted control fields. If two FR runs are made, none of the control fields from the first file will be retained in the final file.

The revised file will contain only the FFT, the logic statements for the report 'FR', and the data records. Before the user processes his new file, he must review and modify, as required, his old logic statements, queries, RITs, tables, and subroutines to ensure compatibility with his new file design. The logic statements must be recompiled and added to the logic statement library (report 'FR' should be deleted). Other materials must be recompiled, as required, and placed on the appropriate library.

4.1 FR Description

The user structures an FFT describing the file in its revised format. He then executes the procedure XFR, identifying both old and new files, field name changes, and the FR output and processing options. This procedure causes execution of the control program FR. This program calls the routines to process the user's control deck, the old FFT, and the new FFT; it also produces the logic statements that may be compiled and executed through linkage to the File Maintenance (FM) component.

After identifying and processing the control options, FR will develop a name change table from the remaining input

parameters. Another table will be developed from the field description records in the revised FFT.

Having accomplished this initialization, File Revision will read the current FFT and generate, for each set type, an FMS logic statement. Any Secondary Indexing Specification must be provided by the user when the new FFT is structured. The data required to generate indexing information will be provided by FM.

FR will build a data set containing FMS control records and the generated logic statements. The data set or selected portions of it may be punched and/or printed. The user may use the FR-generated logic statements, with or without modification, in subsequent independent FM executions.

Additionally, FR will print an FFT comparison. This comparison will indicate the field name, size, storage mode, and set number for each field in both the current and revised files. It will also annotate all changes with the appropriate remarks.

At the successful completion of FR, FM will be executed to compile the generated logic statements for report 'FR' and will execute them to generate the revised file. The old file data records provide the transactions to the FM generation phase.

4.2 Processing Flow

Figure 3 illustrates the processing flow of the File Revision process.

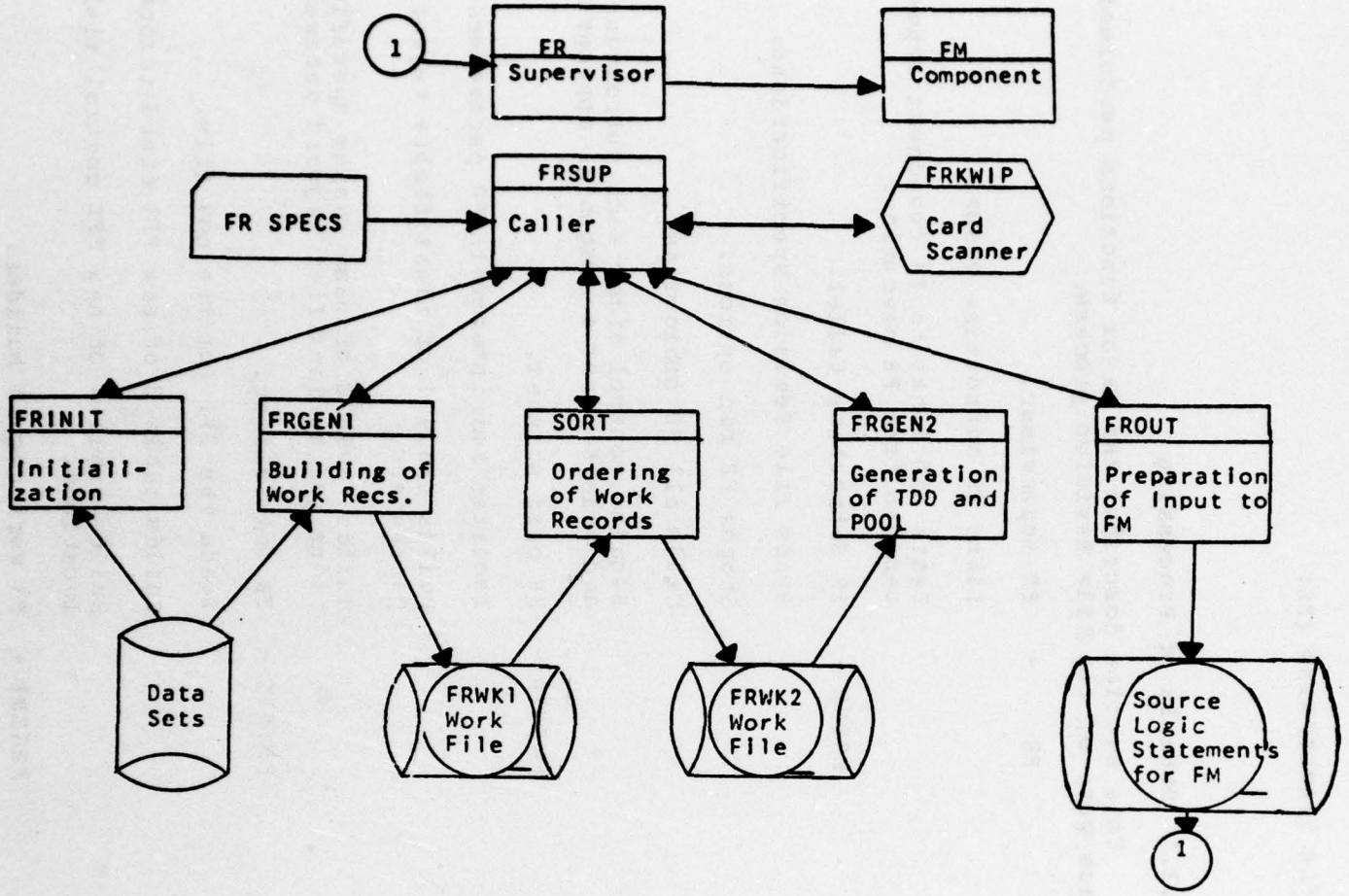


Figure 3. File Revision Process

4.2.1 Details of Processing

This section describes the major functions performed by each pass of the File Revision process.

FR - FR Supervisor.

Links FR subroutine FRSUP.

Tests and links to FM component dependent on results from FR execution.

FRSUP - FR Subroutine Caller.

Reads File Revision specifications.

Stores FR run options.

Calls all FR subroutines.

Regains control after each subroutine and prints error messages as appropriate.

FRKWIP - FR card scanner.

Isolates and interprets FR parameters.

Builds table 1 of identifiable keyword parameters.

Builds table 2 of name change specifications and unidentified keyword parameters.

FRINIT - FR Initializer.

Reads the FFT for the new file.

Builds table 3 of new FFT field information.

Builds table 4 of new FFT control field locations.

FRGEN1 - FR work record builder.

Combines old and new specifications for related data fields.

Builds 28-character records containing field name, size, set, and storage mode.

Writes these records on to the file 'FRWK1', which is then sorted by old set number, new set number, and old field location within set.

FRGEN2 - FR Generator.

Reads the sorted work file 'FRWK2'.

Generates TDD control card images to describe the old data file which is to be used by FM as a data source.

Generates POOL control card images to perform the required movement of data from the old to the new file.

Outputs TDD cards and POOL statements on separate data sets, for use by 'FROUT'.

Prints an analysis of the FR action to be performed with each field.

FROUT - FR output subroutine.

Combines FMS and library action control card images with the TDD and POOL generated by 'FRGEN2'.

Writes out data sets containing control card information to perform logic statement compilation, library action, and file generation.

Prints and/or punches the generated logic statements as required.

Returns control to FRSUP which will in turn pass control to the supervisor, FR, to link and execute the File Maintenance (FM) component.

4.3 FR Input

The following paragraphs describe input data sets and control cards.

4.3.1 Old Data File

The old 360 FFS data file (the file to be revised) may be either a SAM, an ISAM or a VSAM data file. The file organization is identified by using the 'SAM', 'ISAM' or 'VSOLDf' parameters on the XFR execute card.

As the old data file is used as the transaction source to the generate phase of FM, an additional parameter 'TRANTYP' must be used to identify the file organization. This parameter must equal SAM if the old data file is SAM, or VSAM if the old data file is VSAM. If old data file is ISAM, the TRANTYP parameter will default to ISAM.

4.3.2 New File Format Table

Before running File Revision, the new FFT must be structured and must reside on direct access storage as either an ISAM or VSAM data set. The new FFT is sequential organization. The new FFT is identified by using the 'NEWFFT' parameter on the XFR execute card for an ISAM FFT or the 'VSDSN' parameter for a VSAM FFT. The FFT name need not be the same as the old file name.

4.3.3 Card Input

File Revision control cards are free format and, with the exception of the field name change cards, use keywords to identify parameters. Since the field name change information is not identified by keywords, all of the field name change information must follow the keyword-identified information in the user's input deck. Furthermore, no extraneous information, such as comments, may be punched on any of the user's control cards, since FR will interpret this information as field name change information.

The sequence of keyword-identified information will be: KEYWORD, VALUE, KEYWORD, VALUE, etc., with no fixed format or parameter sequence requirement. The user may enter multiple parameters after a single keyword if the parameter list is enclosed in parentheses. Any number of blanks may separate keywords and values from their connecting equal sign; e.g., KEYWORD##=#VALUE. Any number of blanks and/or commas may separate keyword/value pairs; e.g., FILE = TESTA,NEWFILE = TESTB PRINT = ALL.

Special characters other than the comma, equal sign and parenthesis are not valid in File Revision control statements.

Processing options such as PUNCH, PRINT, etc., and file name identifiers are keywords. All other words found in the keyword positions are assumed to be old field names, and the words following them are assumed to be the new names.

Valid input parameters are listed in the following paragraphs.

OLDFILE or FILE specifies the name of the data file to be converted. This parameter is required.

NEWFILE identifies the revised file name. If the file name does not change, this parameter is unnecessary. If this parameter is not specified, the old file name will be used as the file name for the revised file.

The PRINT parameter is optional. PRINT causes the printing of the generated logic statements. It may be followed by the word ALL to cause the printing of all the statements or by the desired set numbers. If the set numbers are used and more than one set is desired, they must be enclosed in parentheses and separated by commas; e.g.,

PRINT = (1, 4, 7, 23)

The PUNCH parameter is optional. PUNCH causes the punching of the generated logic statements. The operands which may be used are the same as those for the PRINT parameter.

NOGO is an optional parameter followed by FM, indicating that the File Revision job should terminate just before execution of File Maintenance.

Field name changes are input in the form -

old-field-name = new-field-name

and must follow all keyword-identified parameters.

4.4 FR Output

The following paragraphs describe the output of File Revision.

4.4.1 Revised File

If no errors are detected by the FR program (and the user has not selected the NOGO=FM option), a revised file will be produced. This file will always be a sequential file, and unless the user specifies the device type by using the 'NEWUSAM' parameter on his execute card, it will reside on the device type specified by the default parameters in the FMSAMOUT DD statement. The new file will consist of the FFT, the logic statements that were created by FR to create the file, and the data records. The old logic statements will not be carried over to the new file. The file name

will either be the same as the old file, or the same as the user specified in his FR control deck, using the 'NEWFILE' parameter. The new file will be cataloged. If the old file was sequentially organized and its name was used for the new file, it will be uncataloged. Otherwise, the old file will remain cataloged. The block size of the new file will be the same as the old file unless the user specifies a new block size using the BSZNEWF symbolic parameter of the XFR procedure.

4.4.2 Index Data Set

If indexes were specified for the revised data file, an Index Data Set will be created. The parameters necessary for creating this data set must be provided by symbolic parameters of the XFR procedure.

4.4.3 Printer Output

FR produces three types of printer output.

The first list consists of a print-back of the user's control deck, with appropriate error comments.

The second printer output consists of a comparison of the old and new FFTs, with field changes indicated and any illegal field changes noted.

The third printer output consists of a list of the generated logic statements that the user requested to be printed by using the PRINT parameter in his control deck. One logic statement will be created for each old set, provided all the fields in a given set were not deleted. They will be printed in set sequence.

Following the FR printouts, the normal FM printouts will appear, showing the compiled logic statements, transaction processing errors which would indicate deletion of a set, and confirmation messages.

4.4.4 Punched Output

FR will punch the generated logic statements for the sets requested in the PUNCH parameter.

4.5 FR Examples

The following examples illustrate typical uses of File Revision.

4.5.1 Sample Revision with Two Name Changes

```
FILE = TESTA, NEWFILE = TESTB
              PUNCH=(4, 7, 9)
              PRINT=ALL

ARRTIM = ATA,
NUMBR = FLTNBR
```

The current file name is TESTA, and the revised file is to be TESTB. The user specifies that the logic statements and FM control records, which are generated for current file set numbers 4, 7 and 9, are to be punched. All the generated logic statements are to be printed. The names of the two current file fields ARRTIM and NUMBR are to be changes to ATA and FLTNBR in the revised file.

4.5.2 File Structure, Revision, and Maintenance

An illustrative example of a file revision run can be found in Vol. VIII, Job Preparation Manual.

4.6 File Revision, Special Considerations

File Revision typically involves adding or deleting of data fields and/or periodic sets. It may also involve changes in mode, size, or location of data fields. These actions will, in many cases, change the relative location of data elements within a set. Although some existing

retrievals and report instruction tables may still be valid, it is recommended that the user recompile them and store them on the appropriate library to preclude production of erroneous output. File Maintenance logic statements must be recompiled and stored on the revised file. This will preclude inadvertent destruction or degradation of the revised file.

In order to revise a segmented SAM data file, a definite process must be followed. Each segment of a segmented data file must be revised in chronological order based on Record ID's. If the Record control field changes, the order should be based on the new Record control fields expected. In general, revising the Record control fields is not advised.

The first step should be creation of the FPT. The execution of FR should be done with the NOGO option allowing storage of the FR logic statements. The third step would involve a SAM segment generation run with a segment control record to define the segment boundary for the first segment. The FPT and Logic statements on the ISAM or VSAM FPT must be used for each generation of a segment. Each segment generation must include a segment control card.

DISTRIBUTION

CCTC CODES	COPIES
C124 (Reference)	2
C124 (Record Copy)	1
C240 (COR for CSC)	20
C300	1
C310	1
C311	1
C312	1
C313	1
C314	1
C315	1
C316	1
C317	1
C320 (Training)	5
C321	1
C322	1
C323	1
C324	1
C325	1
C340	5
C341 (Maintenance Contractor)	8
C341 (Stock)	50
C700	1
C702	1
C703	1
C705	1
C710 (Tech Ser Support)	2
C720	1
C730	2

DCA CODES	
C100	1
C110	1
C205	1
C530	1
C600	1

EXTERNAL	COPIES
Director of Administrative Services, Office of the Joint Chiefs of Staff Attn: Chief, Personnel Division, Room 2A944, The Pentagon Washington, D.C. 20301	1
Director for Personnel, J-1, Office of the Joint Chiefs of Staff, Attn: Chief, Data Service Office, Room 1B738C, The Pentagon, Washington, D.C. 20301	1
Director for Operations, J-3, Office of the Joint Chiefs Staff, Attn: Chief, Data Processing Division, Room 2C869, The Pentagon, Washington, D.C. 20301	1
Director for Operations, J-3, Office of the Joint Chiefs of Staff, Attn: P & AD, Room 2B870, The Pentagon, Washington, D.C. 20301	1
Director for Operations, J-3, Office of the Joint Chiefs of Staff, Attn: Deputy Director for Operations (Reconnaisance and Electronic Warfare) Room 2D921, The Pentagon, Washington, D.C. 20301	1
Director for Logistics, J-4, Office of the Joint Chiefs of Staff, Room 2E828, The Pentagon, Washington, D.C. 20301	1
Chief, Studies Analysis and Gaming Agency, Attn: Chief, Force Analysis Branch, Room 1D928A, The Pentagon, Washington, D.C. 20301	1
Automatic Data Processing, Liaison Office, National Military Command Center, Room 2D901A, The Pentagon, Washington, D.C. 20301	1
Automatic Data Processing Division Supreme Headquarters Allied Powers, Europe Attn: SA & P Branch, APO New York 09055	1
Defense Civil Preparedness Agency, Computer Systems Division, Room 3D317, The Pentagon, Washington, D.C. 20301	1
Director, Defense Communications Agency, Office of MEECN System Engineering, Attn: Code 960T, Washington, D.C. 20301	1
Director, Defense Communications Engineering Center, Hybrid Simulation Facility, 1860 Wiehle Avenue, Reston, VA 22070	1

EXTERNAL	COPI
Commander, Joint Technical Support Activity Attn: Chief, Software Operations Division 1860 Wiehle Avenue, Reston, VA 22070	1
Director, Defense Intelligence Agency Attn: DS - 5C2 Washington, DC 20301	5
Commander-in-Chief, Pacific, Attn: J6331, FPO San Francisco, 96610	1
Commander-in-Chief, Europe, Attn: Chief, EUCOM ADP Systems Office (ECADP) APO New York 09128	1
Commander-in-Chief, US Army Europe and Seventh Army Attn: ODCS, OPS APO New York 09403	1
Commanding General, US Army Forces Command, Attn: Data Support Division, Building 206, Fort McPherson, GA 30303	1
Commander, Fleet Intelligence Center, Europe, Box 18, Naval Air Station, Jacksonville, FL 32212	1
Commanding Officer, Naval Air Engineering Center, Ground Support Equipment Department, SE 314, Building 76-1, Philadelphia, PA 19112	1
Commanding Officer, Naval Security Group Command, 3801 Nebraska Avenue, NW, Attn: GP22, Washington, DC 20390	1
Commanding Officer, Navy Ships Parts Control Center Attn: Code 712, Mechanicsburg, PA 17055	1
Headquarters, US Marine Corps, Attn: System Design and Programming Section (MC-JSMD-7) Washington, DC 20380	1
Commanding Officer, US Army Forces Command Intelligence Center, Attn: AFIC-PD, Fort Bragg, NC 28307	1
Commander, US Army Foreign Science and Technology Center Attn: AMXSJ-CS, 220 Seventh Street, NE, Charlottsville, VA 22212	1

EXTERNAL	COPIES
Commanding Officer, US Army Security Agency, Command Data Systems Activity (CDSA) Arlington Hall Station Arlington, VA 22212	1
Commanding Officer, US Army Security Agency Field Station - Augsburg, Attn: IALADP, APO New York 09458	1
Commander, Fleet Intelligence Center, Atlantic, Attn: DPS, Norfolk, VA 23511	1
Commander, Fleet Intelligence Center, Pacific, Box 1275 FPO San Francisco, 96610	1
Air Force Operations Center, Attn: Systems Division (XOOCSC) Washington, DC 20301	1
Commander, Armed Forces Air Intelligence Training Center, Attn: TINI-MIT, Lowry AFB, CO 80230	1
Commander, Air Force Data Services Center, Attn: Director of System Support, Washington, DC 20330	1
Commander-in-Chief, US Air Forces in Europe, ATTN: ACDI APO New York 09332	1
Commander, USAF Tactical Air Command, Langley AFB, VA 23665	1
Commander, Space and Missile Test Center, Attn: (ROCA) Building 7000, Vandenberg, AFB, CA 93437	1
Naval Air Systems Command, Naval Air Station, Code 13000, Jacksonville, FL 32212	1
Commanding General, US Army Computer Systems Command, Attn: Support Operations Directorate, Fort Belvoir, VA	1
Defense Documentation Center, Cameron Station, Alexandria, VA 22314	12
TOTAL	171

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: CSM UM 15-78, Volume II ✓	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): NMCS Information Processing System 360 Formatted File System (NIPS 360 FFS) - Users Manual Vol II - File Structuring (FS) Vol. I - AD59033 / Vol. 4 - AD58957	5. TYPE OF REPORT & PERIOD COVERED / 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s)	8. CONTRACT OR GRANT NUMBER(s): DCA-100-77-C-0065 ✓
9. PERFORMING ORGANIZATION NAME AND ADDRESS: International Business Machines, Corp. ✓ Rosslyn, Virginia	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: National Military Command System Support Center The Pentagon, Washington, D.C. 20301	12. REPORT DATE: 1 September 1978 / 13. NUMBER OF PAGES: 76
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Copies of this document may be obtained from the Defense Documentation Center, Cameron Station, Alexandria, Virginia 22314.
This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

AD59033 AD58946 AD58966

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This volume focuses primarily on detailed instructions for effectively using the File Structuring and File Revision Components. A brief introduction to these system components and a description of expected output are given.

This document supersedes CSM UM 15-74, Volume ~~II~~ 2.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)